1. A method for detecting attempted intrusions in a database application, the method comprising:

monitoring for an SQL statement, said SQL statement executable in said database application and intended to exploit a vulnerability;

actuating said SQL statement to discover an atomic SQL command;

analyzing said atomic SQL command against a pre-defined set of detection rules.

2. The method according to claim 1, wherein said vulnerability is a buffer overflow in a SQL procedure.

3. The method according to claim 1, wherein said vulnerability is a buffer overflow in a call from SQL to an operating system function.

4. The method according to claim 1, wherein said vulnerability is an attempt to escalate privileges of a user in said database application.

5. The method according to claim 1, wherein said vulnerability is an attempt to escalate privileges within an operating system.

6. The method according to claim 1, wherein said vulnerability is an attempt to insert an invasive SQL statement into a parameter of stored procedures.

7. A method for detecting an anomalous command in a database application, the method comprising:

actuating said database application in order to discover a form of a set of authorized SQL statements and commands and to discover appropriate parameters for said statements and commands;

generating a rule set of said discovered form of said authorized SQL statements;

monitoring for SQL statements executable in said database application which do not match said generated rule set of forms of authorized SQL statements.

8. The method according to claim 7, wherein said anomalous command is a SELECT statement.

9. The method according to claim 7, wherein said anomalous command is an UPDATE statement.

10. The method according to claim 7, wherein said anomalous command is an INSERT statement.

11. The method according to claim 7, wherein said anomalous command is a DELETE statement.

12. The method according to claim 7, wherein said anomalous command is a call to a stored procedure.

13. The method according to claim 7, wherein said anomalous command is a batch script.

14. A method for detecting attempts to access a database application from invalid sources, the method comprising:

actuating said database application in order to discover a normal set of authorized SQL sources;

generating a rule set of characteristics of connecting at least one of said normal set of SQL sources;

monitoring for SQL statements executable in said database application which do not match said generated rule set of valid forms for authorized SQL statements.

15. The method according to claim 14, wherein a characteristic of said rule set is based on a location of an SQL source.

16. The method according to claim 14, wherein a characteristic of said rule set is based on a network address of an SQL source.

17. The method according to claim 14, wherein a characteristic of said rule set is based on a host name of an SQL source.

18. The method according to claim 14, wherein a characteristic of said rule set is based on a domain name of an SQL source.

19. The method according to claim 14, wherein a characteristic of said rule set is based on a time of activity of an SQL source.

20. The method according to claim 14, wherein a characteristic of said rule set is based on an application name of an SQL source.

21. The method according to claim 14, wherein a characteristic of said rule set is based on a behavior of an SQL source.

22. A method for detecting unauthorized activity in a database application, the method comprising:

monitoring for SQL statements executable in said database application and intended to perform activities not authorized by an SQL source;

actuating each discrete database event;

analyzing each event against a pre-defined set of detection rules.

23. The method according to claim 22, wherein said unauthorized activity is accessing data for which said SQL source has not been granted privileges.

24. The method according to claim 22, wherein said unauthorized activity is accessing data not using an authorized method.

25. The method according to claim 22, wherein said unauthorized activity is accessing data in a data dictionary not using an authorized method.

26. The method according to claim 22, wherein said unauthorized activity is interfering with auditing settings.

27. The method according to claim 22, wherein said unauthorized activity is interfering with audit records.

28. The method according to claim 22, wherein said unauthorized activity is modifying configuration settings.

29. The method according to claim 22, wherein said unauthorized activity is modifying security settings.

30. The method according to claim 22, wherein said unauthorized activity is a use of an unauthorized tool to attempt to access said database application.

31. A method for detecting activity designed to breach security of a database application, the method comprising:

monitoring for discrete events executable in said database application and intended to breach a security mechanism associated with said database application;

actuating each discrete database event;

analyzing said database events against a pre-defined set of detection rules.

32. The method according to claim 31, wherein said activity is a brute-force guessing of usernames in said database application.

33. The method according to claim 31, wherein said activity is the brute-force guessing of usernames and passwords for default accounts in said database application.

34. The method according to claim 31, wherein said activity is the brute-force guessing of usernames and passwords for well-known accounts in said database application.

35. The method according to claim 31, wherein said activity is the scripting of password guessing against the database application.

36. A method for detecting suspicious activity in a database application, the method comprising:

monitoring for SQL statements executable in said database application which contain characteristics indicative of an attack;

actuating each batch statement in order to discover atomic SQL commands;

analyzing said atomic SQL commands against a pre-defined set of rules to identify said suspicious activity.

37. The method according to claim 36, wherein said suspicious activity is a use of comments within an SQL statement.

38. The method according to claim 36, wherein said suspicious activity is a use of a UNION keyword within an SQL statement.

39. The method according to claim 36, wherein said suspicious activity is a use of a keyword designed to suppress auditing data.

40. A method for detecting use of keywords to suppress auditing of attacks in a database application, the method comprising:

monitoring for SQL statements that contain a keyword, where said keyword results in audit data being suppressed;

detecting a suppressed SQL statement;

detecting a conclusion of said suppressed SQL statement;

determining that no execution of said keyword designed to suppress said SQL statement actually occurred.

41. The method according to claim 40, further comprising a use of passwords designed to cause an auditing system to suppress text of said SQL statement and masking malicious activity.

42. A host-based intrusion prevention method for blocking attacks on database applications, the method comprising:

detecting an attack occurring through a session with said database application;

identifying a source of said attack;

implementing a method of stopping said attack source;

implementing a method of preventing further attacks from said attack source.

43. The method according to claim 42, wherein said method of stopping said attack source is killing a user connection of said attack source.

44. The method according to claim 42, wherein said method of stopping said attack source is sending a reset to said attack source.

45. The method according to claim 42, wherein said method of stopping said attack source is blocking a SQL command.

46. The method according to claim 42, wherein said method of stopping said attack source is intercepting and filtering a SQL command.

47. The method according to claim 42, wherein said method of stopping said attack source is throwing an exception.

48. The method according to claim 42, wherein said method of preventing further attacks is disabling an account from being used.

49. The method according to claim 42, wherein said method of preventing further attacks is killing any future attempts from said attack source.

50. A method for detecting attempts to inject SQL into a database application, the method comprising:

monitoring for SQL statements executable in said database application and intended to run queries not designed to be run by a middle-tier application;

analyzing said SQL statement's identifying characteristics indicative of SQL injection;

implementing an action upon detection of identifying characteristics indicative of SQL injection.

51. The method according to claim 50, wherein said action is causing a security alert to be fired.

52. The method according to claim 50, wherein said action is causing the SQL statement to be blocked.

53. A method for detecting attempts to inject SQL into a database application, the method comprising:

listening to SQL queries executable on said database application for a determined period of time;

tokenizing SQL statements into standard forms;

recording a combination and an order of tokens expected;

analyzing SQL statements received later to identify those that do not conform to said expected combination of tokens.

54. A method for detecting malicious activity in a database application, the method comprising:

listening to SQL queries executable on said database application;

analyzing SQL statements by applying regular expressions to detect vulnerabilities;

sending alerts when an SQL statement matching a regular expression is discovered.

55. The method according to claim 54, wherein said regular expression is designed to detect a buffer overflow in a call from SQL to a built-in database function.

56. The method according to claim 54, wherein said regular expression is designed to detect a buffer overflow in a call from SQL to an operating system function.

57. The method according to claim 54, wherein said regular expression is designed to detect an attempt to escalate privileges of a user in said database application.

58. The method according to claim 54, wherein said regular expression is designed to detect an attempt to insert an SQL statement into a parameter of stored procedures.

59. The method according to claim 54, wherein said regular expression is designed to detect an attempt to escalate privileges of a user in an operating system.

60. A method for detecting activity which may result in cross-site scripting vulnerabilities, the method comprising:

monitoring for SQL statements executable in said database application;

actuating each batch statement in order to discover atomic SQL commands;

examining an atomic SQL command for HTML tags.

61. The method according to claim 60, wherein said atomic SQL command contains an HTML tag.

62. The method according to claim 61, wherein said HTML tag is unencoded.

63. The method according to claim 61, wherein said HTML tag is hex encoded.

64. A method for monitoring all activity for security auditing, the method comprising:

monitoring for an event generated by a database application;

actuating said event;

recording said event.

65. The method according to claim 64, wherein said event being generated comprises an SQL statement.

66. The method according to claim 64, wherein said event being generated comprises failed logins and successful logins.

67. The method according to claim 64, wherein said event being generated comprises incomplete attempts to access said database application.

68. The method according to claim 64, wherein said event being generated comprises DBA activity.

69. The method according to claim 64, wherein said event being generated comprises changes to a configuration.

70. The method according to claim 64, wherein said event being generated comprises enabling of application roles.

71. The method according to claim 64, wherein said event being generated comprises a method of granting, revoking, or denying permissions or privileges.

72. The method according to claim 64, wherein said event being generated comprises a utility event.

73. The method according to claim 72, wherein said utility event is a backup command.

74. The method according to claim 72, wherein said utility event is a restore command.

75. The method according to claim 72, wherein said utility event is a bulk insert command.

76. The method according to claim 72, wherein said utility event is a BCP command.

77. The method according to claim 72, wherein said utility event is a DBCC command.

78. The method according to claim 64, wherein said event being generated comprises a server shutdown.

79. The method according to claim 64, wherein said event being generated comprises a pause.

80. The method according to claim 64, wherein said event being generated comprises a start-up.

81. The method according to claim 64, wherein said event being generated comprises an audit event.

82. The method according to claim 81, wherein said audit event is an add audit command.

83. The method according to claim 81, wherein said audit event is a modify audit command.

84. The method according to claim 81, wherein said audit event is a stop audit command.

85. The method according to claim 64, wherein said event being generated comprises use of extended stored procedures.

86. A method for providing exceptions to security alerts, the method comprising:

monitoring for events generated by a database application;

filtering alerts raised that match a defined set of rules;

passing alerts not matching a normal definition of said defined set of rules.

87. The method according to claim 86, wherein said defined set of rules comprises values for each field collected for each event.

88. The method according to claim 86, wherein said filtering is matched by comparing values of each field with values defined in an exception.